Amendments to the Claims:

This listing of claims will replace all prior versions of the claims in the present application:

Listing of Claims:

1. (Currently Amended) A computer~~ized~~ implemented method of watermarking a software object, wherein the computer performs the following functions comprising ~~the steps of~~:

(a) determining a watermark by the computer;

(b) determining an input sequence; and

(c) storing the watermark in an execution state of the software object within a memory, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer which examines the execution state of the software object when the software object is being run with the input sequence.

2. (Currently Amended) The computer implemented method as claimed in claim 1, wherein the software object is a program or a piece of a program.

3. (Currently Amended) The computer implemented method as claimed in claim 1, wherein the watermark is detectable in the execution state of the software object formed by the current values held in at least one of:

a) at least one stack;

b) at least one heap;

c) at least one data register; and

d) at least one global variable;

of the software object.

4. (Currently Amended) The computer implemented method of claim 1 or 2 or 3, wherein the watermark is stored in the execution state of the software object whereby the input sequence is constructed which, when fed to an application of which the software object is a part, will make the software object enter a second state which is a representation of the watermark, the representation being validated or checked by examining the execution state of the software object.

5. (Currently Amended) The computer implemented method of claim 1, wherein the watermark is embedded in an execution trace of the software object whereby, as an input determined during the watermarking process is fed to the software object, an address/operator trace is monitored and, based on a property of the trace, the watermark is extracted.

6. (Currently Amended) The computer implemented method of claim 1 or 2 or 3, wherein the watermark is embedded in a topology of a dynamically built graph structure.

7. (Currently Amended) The computer implemented method of claim 6, wherein the dynamically built graph structure is detectable in a data structure of the program.

8. (Currently Amended) The computer implemented method of claim 1, or 2 or 3 further comprising the step of:

(c) building a computerized recognizer operable to examine the execution state of the software object when run with an input sequence and indicate whether the watermark is detectable in the execution state of the software object.

9. (Currently Amended) The computer implemented method of claim 8, wherein the computerized recognizer is a function adapted to identify and extract the watermark from all other dynamic structures on a heap or stack.

10. (Currently Amended) The computer implemented method of claim 8, wherein the watermark incorporates a marker that will allow the computerized recognizer to recognize it easily.

11. (Currently Amended) The computer implemented method of claim 8, wherein the computerized recognizer is retained separately from the program and whereby the computerized recognizer inspects the execution state of the program.

12. (Currently Amended) The computer implemented method of claim 8, wherein the computerized recognizer is dynamically linked with the program when it is checked for the existence of a watermark.

13. (Currently Amended) The computer implemented method of claim 1, or 2, or 3, wherein the software object is a part of an application that incorporates tamper-proofing code.

14. (Currently Amended) The computer implemented method of claim 8, wherein the computerized recognizer checks the watermark for a signature property.

15. (Currently Amended) The computer implemented method of claim 14, wherein the signature property is evaluated by testing for a specific result from a hard computational problem.

16. (Currently Amended) The computer implemented method of claim 14, wherein the watermark is embedded in a topology of a dynamically built graph structure, the method including the step of:

(d) creating a number having at least one numeric property which is an index of an embedded dynamically built graph structure, whereby the signature property is evaluated by testing the at least one numeric property.

17. (Currently amended) The computer implemented method of claim 16, wherein the signature property is evaluated by testing whether the number is a product of two primes.

18. (Currently Amended) A computer implemented ~~The~~ method of verifying the integrity or origin of a program, wherein the computer performs the following functions, including ~~the steps of~~:

(a) watermarking the program with a watermark by the computer, wherein the watermark is stored in an execution state of a program within a memory wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable when the program is being run with an input sequence;

(b) building a computerized recognizer, wherein the computerized recognizer is adapted to extract the watermark from dynamically allocated data wherein the computerized recognizer is kept separately from the program; wherein the computerized recognizer is adapted to check for a number.

19. (Currently Amended) The computer implemented method of claim 18, wherein the number is the product of two primes and wherein the number is the

index of an embedded watermark graph in an enumeration of a class of possibly-embedded watermark graphs each having a different topology.

20. (Currently Amended) The computer implemented method of claim 18, wherein the number is derived from a combination of three or more prime numbers.

21. (Currently Amended) The computer implemented method of claim 18 or 19, wherein the program is further adapted to be resistant to tampering, the resistance to tampering being affected by adding tamper-proofing code.

22. (Currently Amended) The computer implemented method of claim 18 or 19, wherein the computerized recognizer checks for the effect of the watermark on an execution state of the program, thereby preserving an ability to recognize the watermark where semantics-preserving transformations have been applied to the program.

23. (Currently Amended) A computer implemented method of watermarking software, wherein the computer performs the following functions including ~~the steps of~~:

(a) embedding a watermark in a string; and

(b) including in the software code that is stored in memory, the code being

~~Executed~~ executed on at least one input determined during the watermarking process that reproduces the string of step (a), and that produces at least one other string when executed with at least one other input; wherein the code is included in the software using a process that inhibits recognition of the string of step (a) by static or dynamic analysis.

24. (Currently Amended) A ~~computerized~~ computer implemented method of watermarking software, which is the computer performing the following functions comprising ~~the steps of~~:

(a) choosing a watermark from a class of graphs having a plurality of members that are stored in memory, each member of the class of graphs having at least one property, the at least one property being capable of being tested by integrity-testing software; and

(b) storing the chosen watermark in the software in a manner that the watermark is detectable and reproduced by a computerized recognizer which examines an execution state of the software when the software is run with an input sequence determined during the watermarking process.

25. (Currently Amended) The computer implemented method of claim 24, wherein the watermark is rendered tamperproof to certain transformations by subjecting the watermark graph to redundant edge insertion.

26. (Currently Amended) The computer implemented method of claim 24, wherein the watermark is a watermark graph including at least one node and wherein each of the at least one node of the watermark graph is expanded into a cycle.

27. (Currently Amended) A computer~~ized~~ implemented method of fingerprinting software, where the computer performs the following functions comprising ~~the steps of~~:

(a) providing a plurality of watermarked programs having a watermark stored in an execution state of a software object for the program by the computer, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory, the watermark being detectable by a computerized recognizer which examines the execution state of the software object as the software object is being run with a particular input sequence.

28. (Currently Amended) The computer implemented method of fingerprinting software as claimed in claim 27 wherein the plurality of watermarked programs each has a number with a common prime factor.

29. (Cancelled)

30. (Currently Amended) A computer readable medium including a program executed on a computer for watermarking a software object, the program including instructions for causing ~~a~~ the computer to:

(a) receive an input sequence through a data communication channel; and

(b) storing a watermark in an execution state of the software object, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer which examines the execution state of the software object when the software object is being run with the input sequence.

31. (Currently Amended) A computer comprising:

a software object;

means to receive an input sequence; and

a watermark stored in an execution state of the software object, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer which examines the execution state of the software object when the software object is being run with a particular input sequence by the computer.

32. (Currently Amended) A computer implemented method of fingerprinting software, wherein the computer performs the following functions comprising ~~the~~

~~steps of~~:

(a) providing a plurality of watermarked programs by the computer, the plurality of watermarked programs being obtained by watermarking each program of the plurality of programs with a watermark, wherein the watermark is stored in an execution state of a program, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable when the program is being run with an input sequence and building a computerized recognizer wherein the computerized recognizer is adapted to extract the watermark from other dynamically allocated data wherein the computerized recognizer is kept separately from the program; wherein the computerized recognizer is adapted to check for a number.

33. (Currently Amended) A computer implemented ~~computerized~~ method of fingerprinting software, wherein the computer performs the following functions, comprising ~~the steps of~~:

(a) providing a plurality of watermarked programs by the computer, the plurality of watermarked programs being obtained by watermarking each program of the plurality of programs with a watermark, the watermark being obtained by embedding a watermark in a string and including in the software code, the code being executed with at least one input determined during the watermarking process in a manner that the string is reproduced and detectable

by a computerized recognizer, and that produces at least one other string when executed with at least one other input.

34. (Currently Amended) A computer implemented method of fingerprinting software, wherein the computer performs the following functions, comprising ~~the steps of~~:

(a) providing a plurality of watermarked programs, the plurality of watermarked programs being obtained by choosing a graph-theoretic property, using the graph-theoretic property to define the class of graphs satisfying the property, choosing a plurality of graphs from the class to be used as watermarks, and embedding each chosen watermark into an execution state of a software program, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer;

(b) providing a computerized recognizer capable of recognizing each of the embedded watermarks by examining the execution state of the software object when it is being executed; and

(c) providing an integrity tester which tests for the satisfaction of the chosen graph-theoretic property in a possibly-modified version of any of the plurality of watermarked programs.

35. (Currently Amended) A computer-readable medium including a program executed on a computer for verifying at least one of the integrity and origin of a program, the program including instructions for:

(a) watermarking the program with a watermark by the computer, wherein the watermark is stored in an execution state of a program, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable when the program is being run with an input sequence;

(b) building a computerized recognizer, wherein the computerized recognizer is adapted to extract the watermark from other dynamically allocated data, wherein the computerized recognizer is kept separately from the program; wherein the computerized recognizer is adapted to check for a number.

36. (Currently Amended) A computer-readable medium including a program executed on a computer for watermarking software, the program including instructions for:

(a) embedding a watermark by the computer in a static string; and

(b) including in the software code, the code being executed on at least one input determined during the watermarking process that reproduces the string of step (a), and that produces at least one other string when executed with at least one other input.

37. (Currently Amended) A computer-readable medium including a program executed on a computer for watermarking software, the program including instructions for:

(a) choosing a graph-theoretic property by the computer, using the graph-theoretic property to define the class of graphs satisfying the property, choosing a graph from the class to be used as a watermark, and embedding the chosen watermark into an execution state of a software program, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer; and

(b) providing a computerized recognizer capable of recognizing the embedded watermark by examining the execution state of the software object when it is being executed;

(c) providing an integrity tester which tests for the satisfaction of the chosen graph-theoretic property in a possibly-modified version of the watermarked program.

38. (Currently Amended) A computer capable of verifying at least one of the integrity and origin of a program, the computer comprising:

an input sequence;

a watermark for watermarking the program, wherein the watermark is stored in an execution state of a program, wherein the execution state is the non-

static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable when the program is being run with the input sequence;

a computerized recognizer, wherein the computerized recognizer is adapted to extract the watermark from other dynamically allocated data wherein the computerized recognizer is kept separately from the program, wherein the computerized recognizer is adapted to check for a number.

39. (Currently Amended) A computer for watermarking software comprising:

(a) a memory for storing data representing a string that has a watermark embedded in it; and

(b) a technique for including in software code by the computer, the code being executed on at least one input determined during the watermarking process that reproduces the string so as to be detectable by a computerized recognizer, and that produces at least one other string when executed with at least one other input.

40. (Currently Amended) A computer comprising:

(a) a watermarked program obtained by choosing a graph-theoretic property, using the graph-theoretic property to define the class of graphs satisfying the property, choosing a graph from the class to be used as a watermark, and embedding the chosen watermark into an execution state of a

software program, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory in a manner that the watermark is detectable by a computerized recognizer;

(b) a computerized recognizer capable of recognizing the embedded watermark by examining the execution state of the software object when it is being executed;

(c) an integrity tester which tests for the satisfaction of the chosen graphtheoretic property in a possibly-modified version of the watermarked program.

41. (Currently Amended) The computer implemented method of claim 1, wherein the watermark is detectable in any portion of the dynamic data state of the software object.

42. (Currently Amended) The computer implemented method of claim 1, wherein the software object is an executable media object.

43. (Currently Amended) The computer implemented method of claim 1, wherein no part of the execution state of the software object in which the watermark becomes detectable is visually or audibly apparent.

44. (Currently Amended) The computer implemented method of claim 8, wherein the computerized recognizer is built concurrently with the watermark and input sequence.

45. (Currently Amended) The computer implemented method of claim 16, wherein said graph is an enumeration of a class of possibly-embedded watermark graphs each having a different topology.

46. (Currently Amended) The computer implemented method of claim 18, wherein the computerized recognizer is built concurrently with the watermark and input sequence.

47. (Currently Amended) The computer implemented method of claim 23, wherein the software code includes a state variable updated during execution of the code and that influences the sting outputted.

48. (Currently Amended) The computer implemented method of claim 31, wherein no part of the execution state of the software object in which the watermark becomes detectable is visually or audibly apparent.

49. (Currently Amended) A computer implemented method of watermarking software, wherein the computer performs the following functions, including ~~the steps of~~:

(a) embedding a watermark in a string, thereby forming a watermark string;

(b) converting the string into executable code and executing the code to constructs a dynamic string by:

(i) receiving an input in the form of at least one input variable;

(ii) defining and updating a string index variable that controls the location of writing to the dynamic string;

(iii) defining a state variable and dynamically determining a state variable so that the value of said state variable in at least one read operation varies depending on which of at least two execution paths was taken by said executable code to reach said operation; and

(iv) dependent on said at least one input variable, string index variable and state variable, writing to the dynamic string; wherein the dynamic string reconstructs the watermark string for at least one input determined during the watermarking process and not for the other inputs.

50. (Currently Amended) A computer implemented method of watermarking a software object, wherein the computer performs the following functions, comprising the steps of:

(a) providing an input sequence by the computer; and

(b) storing a watermark in an execution state of the software object as the software object is being run with the input sequence, the input sequence being fed to an application of which the software object is a part to make the software

object enter a second state which is a representation of the watermark, the representation being validated or checked by examining the execution, of the software object.

51. (Currently Amended) A computer implemented method of watermarking a software object, wherein the computer performs the following functions, comprising ~~the steps of~~:

(a) providing an input sequence by the computer; and

(b) storing a watermark in an execution state of the software object, wherein the execution state is the non-static state of the software object as it is being run on the computer with a particular input sequence, wherein the watermark is stored in the memory as the software object is being run with the input sequence, wherein the watermark is embedded in an execution trace of the software object, and subsequently extracting the watermark by monitoring an address/operator trace as input determined during the watermarking process is fed to the software object.

52. (Currently Amended) ~~The~~ A computer implemented method of varying the integrity or origin of a program, wherein the computer performs the following functions, including ~~the steps of~~:

(a) watermarking the program with a watermark by the computer, wherein the watermark is stored in an execution state of a program, wherein the execution state is the non-static state of the software object as it is being run on

the computer with a particular input sequence, wherein the watermark is stored in the memory as the program is being run with an input sequence in a manner that the watermark is detectable on a computerized recognizer;

(b) building a computerized recognizer, wherein the computerized recognizer is adapted to extract the watermark from dynamically allocated data, wherein the computerized recognizer is kept separately from the program; wherein the computerized recognizer is adapted to check for a number, wherein the computerized recognizer checks for the effect of the watermark on the execution state of the program, thereby preserving an ability to recognize the watermark where semantics-preserving transformations have been applied to the program.

53. (Currently Amended) A computer implemented method of watermarking software, wherein the computer performs the following functions, including ~~the steps of~~:

(a) embedding a watermark in a string by the computer; and

(b) including in the software code, the code being executed on at least one input determined during the watermarking process that reproduces the string of step (a), and that produces at least one other string when executed with at least one other input; wherein the part of the code that manipulates dynamically allocated structures is constricted so that it is computationally difficult to statically determine whether it is safe to perform a transformation of the code incorporating the at least one opaque predicate or variable.

54. (Currently Amended) A computer implemented method of watermarking software, wherein the computer performs the following functions, including ~~the steps of~~:

(a) embedding a watermark in a string; and

(b) including in the software code, the code being executed on at least one input determined during the watermarking process that reproduces the string of step (a), and that produces at least one other string when executed with at least one other input; wherein at least a portion of the code incorporates at least one opaque predicate and the code is constructed so that it is computationally difficult to statically determine whether it is safe to perform a transformation of the code incorporating the at least one opaque predicate.